# Parametric Cost Analysis Using Neural Networks

**By**
**Edwin B. Dean**
**Consultant**
**designforvalue@att.net**

## Abstract

Regression, in various forms, is the tool of choice for parametric cost analysis (PCA). Neural networks are a largely unused analysis tool within the cost community. However, neural networks are an effective alternative to all forms of regression for PCA. This paper will illustrate the use of neural networks for PCA and will highlight similarities and differences with respect to the use of regression for PCA. For example, regression provides a linear or transformed linear approximation of parameters for a given set of data. Neural networks provide a purely nonlinear approximation of the parameters for the data. When a nonlinear equation is desired with regression, the nonlinear transformation is assumed and applied to the data prior to regression. Neural networks find the best nonlinear fit with no prior transformation and no assumptions.

This paper will provide a history of the use of neural networks for cost analysis. This paper will also provide an insight into how neural networks process data and how that process and the associated results differ from those of regression. A method of visual analysis that permits the user to see what is happening will be provided.

Also, a method will be provided that permits a valid comparison of the goodness of fit to a given set of data for all forms of estimating, including linear regression, nonlinear transformed linear regression, neural networks, and all other forms of estimating parameters for PCA.

Attendees will leave with the information necessary to learn more about neural networks and how to get started using them for PCA.

## Neural Networks: How They Work

A neural network is a mathematical entity that simulates the learning capability of the brain. It learns by approximating a set of outputs given a set of inputs.

It has multiple inputs and multiple outputs. The result of the learning process is a model that predicts the desired outputs based upon a set of (input, output) data. A neural network has an input layer, one or more hidden layers, and an output layer. The input layer contains all of the inputs. Each hidden layer and the output layer contain one or more artificial neurons. Each artificial neuron in the output layer corresponds to a particular output. Each artificial neuron has multiple inputs, a threshold, an activation function, and a single output. The activation function is a nonlinear function, such as the inverse tangent or the logistic function, that provides the learning capability.

The learning occurs by applying the (inputs, outputs) data to the network. As each set of input data is applied, the weight of each input to and the level the threshold for each artificial neuron is adjusted to provide a better prediction of the set of outputs. A single application of all of the (input, output) data is called an epoch. Epochs are applied until the inputs provide a reasonable approximation of the outputs.

There are many types of neural networks. The only type addressed in this paper is the backpropagation model originally developed by Werbos (1974) and, later, independently developed by Rumelhart, Hinton, and Williams (1986). This model attempts to reduce the root mean square of the error between the ouputs of the neural network output layer and the output data.

**Neural Network Background of the Author**

In 1963 for his senior Physics project, the author developed an electronic recurrent artificial neural network using sonar as the memory stream. In1965 the author wrote his masters thesis in Mathematics on neural networks. During his tenure with the Naval Ordnance Laboratory, the author applied various types of neural networks for the analysis of various types of data. In 1987 the author began to apply neural networks for the analysis of cost data. From 2001 through 2007, the author applied neural networks to cost data for Galorath, Inc. The author currently uses neural networks for the analysis of cost and the stock market.

**Neural Networks: Prior Use for Cost Analysis**

The primary publications of neural networks applications cost analysis have occurred within the Association for the Advancement of Cost Engineering International (AACEI) community.

The first neural network for cost analysis published papers were McKim (1993a) and McKim (1993b). Both McKim (1993a) and McKim (1993b) introduce the concept of the backpropagation neural network and provide an excellent introduction to the mathematics involved. McKim (1993b) reported that "The neural network trained for this paper was able to predict the cost overruns of ten projects with approximately half the error of more conventional statistical methods."

Today, when one searches on the AACEI library using the keyword "neural network", twenty abstracts appear. Early papers include the following. Creese and Li (1995) estimate the cost of timber bridges using a neural network. Garza and Rouhana (1995) describe the mathematics of the backpropagation neural network and then compare the results of a neural network prediction with regression predictions. Rowings and Sonmez (1996) apply both regression and neural networks for the analysis of labor productivity. Moselhi and Siqueira (1998) use a neural network to estimate the cost of structural steel buildings. Al-Tabtabai, Alex, and Tantash (1999) propose applying subjective data to a neural network to predict preliminary highway construction costs.

Note that neural networks have been and can be used for a number of types of cost analysis. Neural networks also have been and can be applied to finance (Trippi and Turban, ed. (1996)). Note also that neural networks, being models of the way we think human brains may learn and think, allow the input of subjective data as well as hard data.

**An Example of a Neural Network Cost Model**

In order to demonstrate a point, the following neural network cost model is based upon contrived data.

A typical parametric cost model is of the form $y = ax^b$, where $y$ is the cost, $x$ is weight, $a$ is the modeled complexity and $b$ is rate of change of cost with respect to weight. It is a nonlinear model. If we were developing a regression model, we would transform the data and use the equation $\ln(y) = c + b\ln(x)$, where $c = \ln(a)$, to estimate $b$ and $c$. Since the backpropagation neural network can approximate nonlinear equations, we do not have to transform the data, although we could if we wished to do so.

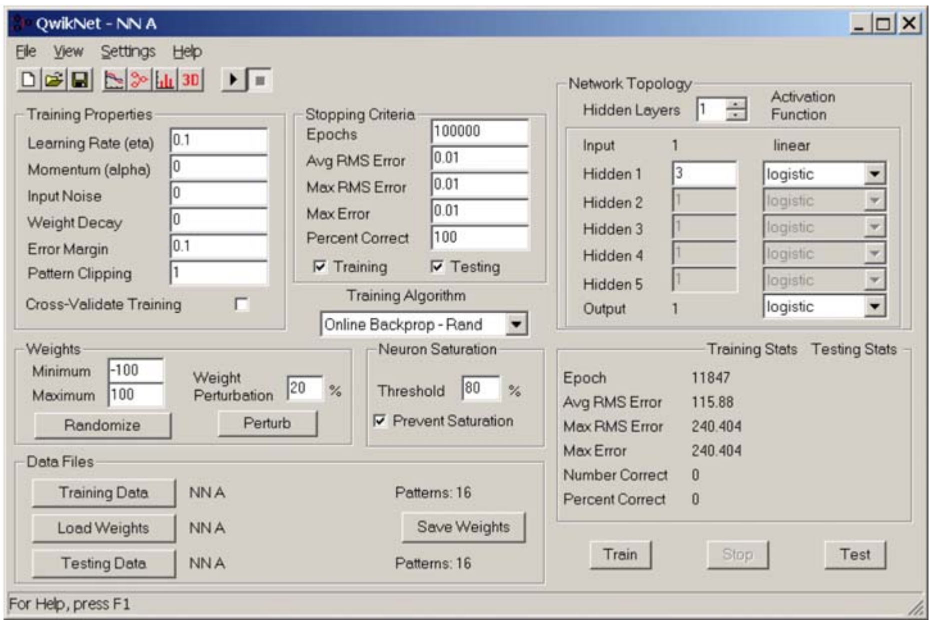
The data for this example was derived by choosing $b$ and $c$ for the equation $\ln(y) = c + b\ln(x)$ and then selecting $b_i$ and $c_i$ for the data from a normal distribution centered on $b$ and $c$ for a given $x_i$ to obtain $y_i$.

At right are the data where $x$ would be the parameter, such as weight, used to estimate y, the simulated cost. The data $z$ are $z = ax^b$ prior to noise being added. Thus $\{(x_i, z_i)\}$ is the set of points of the equation underlying the simulation.

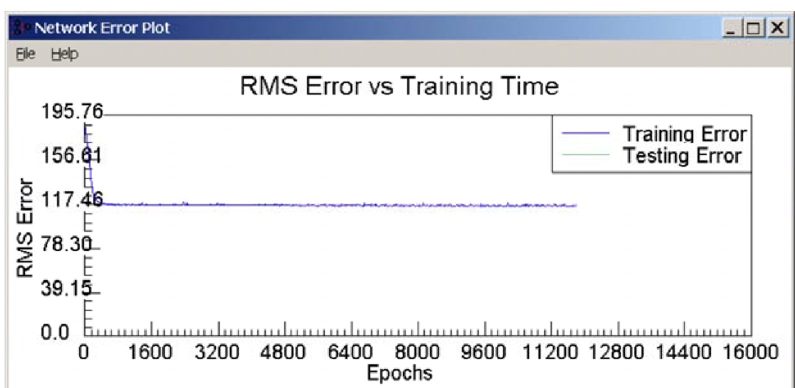
| x | y | z |
|---|---|---|
| 100 | 386.56565 | 185.604698 |
| 108.044801 | 263.437665 | 195.934844 |
| 137.605224 | 135.236046 | 232.077698 |
| 177.879093 | 339.800484 | 277.764282 |
| 218.933064 | 273.434789 | 321.223206 |
| 218.94707 | 432.496933 | 321.237591 |
| 255.354506 | 384.975446 | 357.758064 |
| 280.076661 | 302.904774 | 381.665415 |
| 295.537837 | 592.443618 | 396.294575 |
| 308.377906 | 344.494947 | 408.269787 |
| 343.312423 | 540.576817 | 440.12053 |
| 398.480426 | 410.729949 | 488.51045 |
| 450.550024 | 399.290353 | 532.364486 |
| 497.332603 | 859.150582 | 570.481828 |
| 551.212162 | 574.435695 | 613.072898 |
| 715.503408 | 827.97908 | 735.897214 |

The shareware QuikNet (Jensen, C. (2003)) neural network software was used to run the data in this paper.

As can be seen from the control panel at right, the neural network had one input, weight, and one output, cost, with one hidden layer with three neurons. A logistic function was used as the activation function. Backpropagation with random perturbation was used as the training algorithm. The same data was used for training and testing. The training process was allowed to run for 11,847 epochs. Each epoch is the application of each of the data points.




Root Mean Square (RMS) Error (RMSE) is used as the goodness of fit measure. In the figure below the RMSE rapidly declines and then reaches an almost steady state level at which the training is assumed to be complete.




When using regression and a neural network, the best policy is to have a test data set that is different from the training data set. For this case, in keeping with typical cost model development, "because of the low number of data points", the training data set was used as the test data set as well.

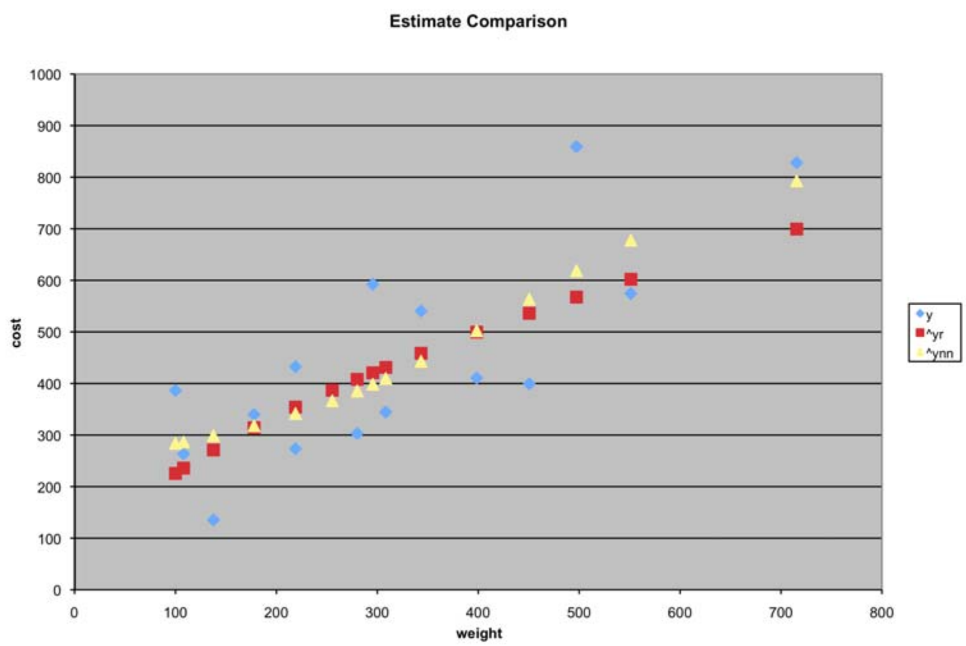
Note that the backpropagation model uses nonlinear optimization and solutions can be caught at sub-optimal points. Thus, training neural networks becomes an art rather than the science of applying regression. The experience of the author is that when using one or two hidden layer backpropagation neural networks with one or two inputs and one or two outputs for the analysis of cost data, this has generally not been a problem. However, the use of a number of subjective data inputs and two or more hidden layers greatly increased the probability of a training session resulting in a suboptimal solution. The larger and more subjective the set of inputs and outputs, the greater the art required in training.

**Analyzing Neural Network Model Results**

When we analyze the results of a regression model of the data, we rely on the well-established methods of statistics. The equations of statistics for regression do not apply to neural networks. For example, the typical goodness of fit for regression is $R^2$. The goodness of fit for neural networks is RMSE.

The author uses two approaches to provide a comparison of regression and neural network results.

The first is visual analysis. For regression, we often plot the equation obtained from the regression and the data points on the same graph. To compare neural network results to regression results, we just add the neural network predicted data points to the same graph as at right.




The cost data points $y$ are diamonds, the regression line data points $\hat{}yr$ are squares, and the neural network predicted data points $\hat{}ynn$ are triangles.

One can easily see that the regression line data points follow the curve $\hat{}y = \alpha x^{\beta}$ where $\alpha$ is the regression estimate of $a$ and $\beta$ is the estimate of $b$. There is no explicit equation for the neural network results, but it is clear from the graph that the neural network prediction curve is similar to that from regression, but is nonlinear.

When there is more than one output, say development cost, production cost, and operations cost, the author creates a separate graph for each output. If there is more than one input, then the author creates a three dimensional graph for each pair of inputs for a given output. This allows one to see the shape of the cost surfaces derived from the input pair. Note that cost surfaces exist for both the neural network and regression results. The visual analysis for multiple inputs and multiple outputs requires a lot of effort but can be very rewarding once the nature of the cost surfaces are determined.

The mathematical basis of the second form of analysis is described by Dean (2008). Consider the points $x_i$, $y_i$, $\hat{y}r_i$, and $ynn_i$ to be dimensions within the vectors $\vec{x}$, $\hat{\vec{y}}$, $\hat{\vec{y}}_r$, and $\hat{\vec{y}}_{nn}$, respectively. A natural geometric measure is the angle between two vectors. The smaller the angle, the more the likeness of the vectors. To compare the regression model with the neural network model, we use this angle as the goodness of fit measure. We calculate the angles $angle(\vec{y}, \vec{y}_r)$ and $angle(\vec{y}, \vec{y}_{nn})$ using the equation

$$angle(\vec{u}, \vec{v}) = \frac{180}{pi} \arccos\left(\frac{\vec{u} \bullet \vec{v}}{\sqrt{\vec{u} \bullet \vec{u}}\sqrt{\vec{v} \bullet \vec{v}}}\right)$$

where $angle(\vec{u}, \vec{v})$ is in degrees, $pi$ is 3.14159…, $\sqrt{w}$ is the square root of $w$, and $\bullet$ is the vector dot product. This equation can be easily implemented in Excel using array formulas.

Applying this formula we have $angle(\vec{y}, \hat{\vec{y}}_r)$= 14.56 degrees and $angle(\vec{y}, \hat{\vec{y}}_{nn})$ = 13.94 degrees.

A noisy and difficult to fit data set was used to display the nonlinearity of the neural network predicted line. As difficult as this data was to fit well, the author has often worked with worse data. As a statistical reference point for the regression goodness of fit, $R^2$= 0.5161. Note that $R^2$ is for the linear fit on the transformed data space. The $angle(\vec{u}, \vec{v})$ is used to compare the predictions on the untransformed data space.

Contrived data was also used in order to see how well the regression model could estimate the parameters of a noiseless equation. The fitted intercept coefficient was 2.7717 vs. 2.0000 for the noiseless equation. The fitted power was 0.5749 vs. 0.7000 for the noiseless equation.

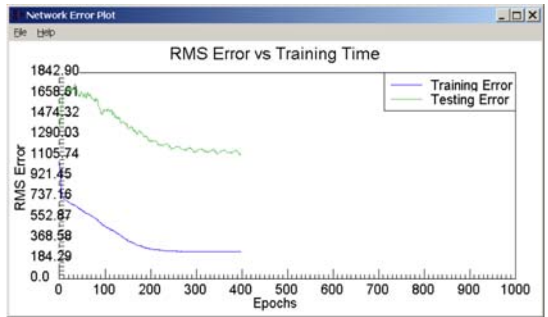
**An Instructive Example**

This example walks you through a real world example of neural network model development. The database has been culled from the Encyclopedia Astronautica Launch Vehicle Summary Data (Wade (1997-2008)) by eliminating all launch vehicles that do not have the desired cost variables and performance parameters. A 5 input, 1 hidden layer with 8 artificial neurons, 2 output neural network (1.8.2) is used to model the development cost and the recurring production cost using total length, core diameter, span, total mass, and maximum range as performance parameters. A logistic activation function is used with a randomly perturbed backpropagation training algorithm.

The results are compared to the results of a 5 parameter multivariate regression for each cost using the same performance parameters. Graphing the costs against the performance parameters indicates that log transformed data should be used for the regressions. The neural network inputs are the untransformed performance parameters.

Note that $R^2$on the transformed data space for the regressions cannot be projected onto the untransformed data space. The desire is to compare both the regressions and the neural network results on the untransformed data space. Thus, the results of the regression are transformed back to the untransformed data space of cost for comparison. By using the angle between each cost data vector and its predicted cost data vector for the both regressions and the neural network we obtain an apples to apples comparison.

The neural network does not adjust the data for the means. Regression uses mean-adjusted data. We can relate the angle used here back to statistics by noting that the multivariate correlation coefficient, $R$, from the regression is the cosine of the angle between the mean-adjusted output data vector and the predicted mean-adjusted output data vector (Wickens (1995)). The cosine of the angle used here can be thought of as the multivariate correlation coefficient for the non mean-adjusted data. By using this angle we can compare the results of all estimating means directly on the nontransformed (original) data space.




For this example we split the data into a training data set and a test data set. We train the neural network on the training data set and use the test data set to monitor the training process and to test how well the neural network generalizes to similar but different data.

The error reduction of the training process, above right, shows the stabilization of the error of the test set above and the training set below. If allowed to train further, the error of the test set rises as the network over trains on the training data at the expense of the error over the test set.

The results at right follow.

| Goodness of Fit (degrees) | Multivariate Regression Training Data | Neural Network Training Data | Multivariate Regression Test Data | Neural Network Test Data |
|---|---|---|---|---|
| Development Cost | 31.6 | 14.2 | 72.4 | 8.4 |
| Recurring Production Cost | 11.7 | 14.9 | 89.8 | 34.5 |

**Potential Uses of Neural Network Project Models**

Because neural networks are multi-input, multi-output estimators, they have many potential uses in project management. The AACEI papers cited above point to an array of potential uses. The theme of the 2009 NASA Cost Estimating Symposium was "Probabilistic Joint Cost and Schedule Estimating: YES WE CAN." A neural network is not probabilistic, however, it is a natural tool for developing a joint cost and schedule model. Above, the author has alluded to the possibility of jointly estimating development, production, and operations cost from a set of input parameters, some of which could be subjective in nature. What other project measures could be combined into this list of outputs? Let your project knowledge be your guide.

**Getting Started with Neural Networks**

The AACEI references in this paper provide adequate information to understand the mathematics underlying the backpropagation neural network. They may be obtained at the AACEI website (AACEI). Search for them at the bookstore using the keyword “neural network”. AACEI members can download the articles for free.

Jones and Hoskins (1987) and Hecht-Nielsen (1992) do an excellent job of explaining the inner workings of the backpropagation neural network. For a detailed introduction to several of the more popular types of neural networks, including backpropagation, study Lippmann (1987). Haykin (1994, 1998) is the comprehensive source for learning about all types of neural networks.

By downloading the shareware program QuikNet (Jensen (2003)), one can begin to get the feel of training a neural network and actually develop models. The author uses this software under Windows Professional 2000 Pro. Hopefully it will work under Windows XP and Windows Vista.

For more sources for learning about neural networks or for more advanced neural network software, perform a web search for "neural network" or "neural network software".

**Summary and Conclusion**

There is a learned art to the development of neural network models. They can be used to develop project measure models including cost and schedule as outputs from a set of input parameters, some of which could be subjective in nature. They facilitate multi-input, multi-output model development. They are inherently nonlinear which opens many modeling possibilities restricted by linear models and related data transformation assumptions. Since they are not based upon statistics, which requires a substantial number of data points for validity, there is no validity restriction on how few data points can be used to develop the model. The use of the angle between data and predicted data vectors provides an intuitive and understandable goodness of fit measure by which to compare the results of all types of models. Engineering managers who are the recipients of the cost estimates are usually familiar with the angle, while there is an excellent chance that they are not familiar with $R$ or $R^2$. Finally, the experience of the author is that properly trained neural network models usually provide a closer match to the data and generalize to test data better than do equivalent regression models.

**References**

AACEI. Association for the Advancement of Cost Engineering International, http://www.aacei.org/.

Al-Tabtabai, H., A. Alex, and M. Tantash (1999). "Preliminary Cost Estimation of Highway Construction Using Neural Networks", *Cost Engineering*, Vol. 41, No. 03 AACE International, Morgantown, WV.

Creese, R. C. and L. Li (1995). "Cost Estimation of Timber Bridges Using Neural Networks", *Cost Engineering*, Vol. 37, No. 05, AACE International, Morgantown, WV.

Dean, E. B. (2008). "The Angle: A Goodness of Fit Metric for Cost Analysis", Proceedings of the ISPA/SCEA 2008 Joint International Conference, 12-16 May 2008, Noordwijk, The Netherlands.

Garza, J. de la and K. G. Rouhana (1995). "Neural Networks Versus Parameter-Based Applications in Cost Engineering", *Cost Engineering*, Vol. 37, No. 02, AACE International, Morgantown, WV.

Haykin, S. (1994, 1998). Neural Networks: A Comprehensive Foundation, Macmillan College Publishing Company, New York, NY.

Hecht-Nielsen, R. (1992). "Theory of the Backpropagation Neural Network", in Neural Networks for Perception, Vol. 2, Harcourt Brace & Co., Orlando, FL. pp. 65-93.

Jensen, C. (2003). QuikNet v2.23, http://qwiknet.home.comcast.net/~qwiknet/.

Jones, W. P. and J. Hoskins (1987). "Back-Propagation: A Generalized Delta Learning Rule", *Byte*, October, pp. 155-162.

Lippmann, R. P. (1987). "An Introduction to Computing with Neural Nets", *IEEE ASSP Magazine*, pp. 4-22.

McKim, R. A. (1993a). "Neural Network Applications to Cost Engineering", *Cost Engineering*, Vol. 35, No. 07, AACE International, Morgantown, WV.

McKim, R. A. (1993b). "Neural Networks and Identification and Estimation of Risk", 1993 AACE International Transactions, P.05, AACE International, Morgantown, WV.

Moselhi, O. and I. Siqueira (1998). "Neural Networks for Cost Estimating of Structural Steel Buildings", 1998 AAEE International Transactions, IT/IM.06, AACE International, Morgantown, WV.

Rowings, J. E. Jr. and R. Sonmez (1996). "Labor Productivity Modeling with Neural Networks", 1996 AACE International Transactions, PROD.01, AACE International, Morgantown, WV.

Rumelhart, D. E, G. E. Hinton, and R. J. Williams (1986), "Learning Representations by Back-Propagating Errors", *Nature*, 323, pp. 533-536.

Trippi, R. R. and E. Turban, ed. (1996). Neural Networks in Finance and Investing, McGraw-Hill, New York, NY.

Wade, M. (1997-2008). Launch Vehicle Summary Data, Encyclopedia Astronautica, http://astronautix.com/data/lvs.xls .

Werbos, P, J. (1974). Beyond Regression: New Tools for Prediction and Analysis in The Behavioral Sciences, PhD Dissertation, Harvard University, Cambridge, MA.

Wickens, T. (1995). The Geometry of Multivariate Statistics, Lawrence Erlbaum Associates, Hillsdale, NJ, USA.